REMARKS

Claims 4 and 6 are amended, no claims are canceled, and no claims are added; as a result, claims 1-9 are now pending in this application.

No new matter has been added through the amendments to claims 4 and 6. Support for the amendments to claim 4 may be found throughout the specification, including but not limited to the specification on page 7 at lines 4-6 and lines 17-18. Support for the amendments to claim 6 may be found throughout the specification, including but not limited to the specification on page 8 at lines 16-18 and 34-37.

Applicants have included with this response copies from various pages, including pages 68-69, 71-73, and 527-531 from the book "Applied Cryptography, Second Edition, Protocols, Algorithms, and Source Code in C." by Bruce Schneier, the book being referenced in the specification of the present application on page 8 at lines 19-22. Comments regarding these various pages are included in the arguments presented below.

§102 Rejection of the Claims

Claims 4 and 8 were rejected under 35 U.S.C. § 102(a) for anticipation by Birdwell et al. (WO99/19822).

Applicants disagree that the Birdwell et al. patent is a § 102(a) prior art reference. Applicants believe that the Birdwell et al. patent is characterized as a reference under 35 U.S.C. § 102(e), and thus believe that Applicants also have the right as provided under 37 C.F.R. 1.131 to swear behind the Birdwell et al. patent. Therefore, Applicants do not admit that the Birdwell et al. patent is prior art, and reserve the right, as provided for under 37 C.F.R. 1.131, to "swear behind" the Birdwell et al. patent. However, Applicants do not believe it is necessary to swear behind Birdwell et al. at this time because claims 4 and 8 are not anticipated by Birdwell et al. for at least the reasons stated below.

Claims 4 and 8 are not anticipated by Birdwell et al. because Birdwell et al. fails to teach each of the elements included in claims 4 and 8.

Claims 4 and 8 include one or more elements not taught by Birdwell et al., and so claims 4 and 8 are not anticipated by Birdwell et al. For example, claim 4 as amended now recites,

> wherein a set of search EMMs is sent to at least a part of the terminals, each search EMM of the set comprising a different dummy key (P_D), each of the dummy keys (P_D) being a key other than a correct service key (P_T) for decrypting an encrypted first key (C_W), and each search EMM being sent to a different terminal or group of terminals.

The subject-matter of claim 4 is novel compared with Birdwell et al. because Birdwell et al. does not disclose a method in which a set of search EMMs is sent of which each search EMM comprises a different dummy key, with each of the dummy keys being a key other than a correct service key for decrypting an encrypted first key, and each search EMM being sent to a different terminal or group of terminals, as recited in now amended claim 4.

It is clear that a "dummy key" equates to a false key, and that a false key is a key that is not a correct key for decrypting any of the encrypted first keys.

Instead, Birdwell et al. discloses that a key generator generates multiple authorization keys for a single data transmission (page 16, l. 2). The server encrypts the session keys with the authorization keys (page 16, l. 18-19). The encrypted session keys are transmitted to the authorized clients (page 16, l. 19-20). Thus, multiple copies of the first keys are encrypted under different service keys, forming a set that is distributed among the terminals.

Because Birdwell et al. fails to teach each of the elements recited in claim 4, claim 4 is not anticipated by Birdwell et al. Because claim 4 is not anticipated by Birdwell et al., the 35 U.S.C. § 102 rejection of claim 4 cannot stand.

Claim 8 depends from claim 4, and so includes all of the elements recited in claim 4. For at least the reasons states above with respect to claim 4, Birdwell et al. fails to teach each of he elements included in claim 8. Therefore, claim 8 is also not anticipated by Birdwell et al., and thus the 35 U.S.C. § 102 rejection of claim 8 cannot stand.

Applicants respectfully request withdrawal of the rejection, and reconsideration and allowance of claims 4 and 8.

§103 Rejection of the Claims

Claims 6 and 9 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Thomson Multimedia EP 0822720A1(hereinafter "Thomson"), and in further view of Birdwell et al.

Applicants respectfully submit that claims 6 and 9 are not obvious in view of the proposed combination of Thomson and Birdwell et al. for at least the reasons stated below.

Claims 6 and 9 are not obvious in view of the proposed combination of Thomson and Birdwell et al. because the proposed combination of Thomson and Birdwell et al. fails to teach or suggest each of the elements included in claims 6 and 9.

Claims 6 and 9 include one or more elements not taught or suggested by the proposed combination of Thomson and Birdwell et al., and so these claims are not obvious in view of the proposed combination of Thomson and Birdwell et al. For example, claim 6 as amended now recites,

> wherein the source signal or the ECMs are encrypted using a multiple-key or secret-sharing cryptographic algorithm having a plurality of different decrypting keys or shares (C_i;P_i) of which any one or any one or more, respectively, is required for decrypting the encrypted source signal or ECMs, respectively, wherein said plurality of different decrypting keys or shares (C_i;P_i) are sent to at least a part of the terminals such that different terminals or groups of terminals receive different keys or shares (C_i;P_i) according to a predetermined distribution.

In contrast, Birdwell et al. discloses a data delivery system in which content is delivered to multiple clients (p. 5, l. 22-23). Data is encrypted prior to transmission (p. 7, l. 3). A content server generates the keying materials used to encrypt the content, and transmits the keying materials ahead of the content to the authorized clients (p. 8, l. 1-3). A key generator creates two tiers of random symmetric keys (p. 8, l. 25). The keys in the first tier are called "session keys" and are used to encrypt the data being served. The keys in the second tier are referred to as authorization keys and are used to encrypt the session key (p. 8, l. 25 – p. 9, l. 2). The authorization keys are preferably distributed to the authorized clients in encrypted format using the authorized clients' public keys of asymmetric key pairs (p. 9, l. 19-20). To trace illicit activity, the key generator generates one or more session keys and multiple authorization keys

for a single data transmission. A key/client associator relates the different authorization keys to different authorized clients (p. 16, l. 1-4). The server encrypts the data with one or more session keys and then encrypts the session keys with the authorization keys. The encrypted session keys are transmitted over the networks to the authorized clients just before the data transmissions (p. 16, l. 17-20). Through surveillance techniques, the illicitly transferred authorization key is discovered (p. 17, l. 3-5).

However, there is thus no disclosure in Birdwell of encrypting the source signal or the ECMs using a multiple-key or secret-sharing cryptographic algorithm having a plurality of different decrypting keys or shares ($C_i;P_i$) of which any one or any one or more, respectively, is required for decrypting the encrypted source signal or ECMs, respectively, as recited in claim 6.

Further, there is no teaching or suggestion of the elements included in claim 6 and missing from Birdwell et al. in Thomson.

In contrast to claim 6, Thomson discloses a system for supplying services to users, which services consist of an item scrambled by control words (column 3, l. 43-46). The system comprises, for each user, a decoder and at least one user card (column 3, l. 46-47). The card contains a circuit making it possible to retrieve the control words from the control words encrypted by an algorithm with key K, the encrypted control words being conveyed to the user card by a message MEC (column 3, l. 53-56). The message contain additional items, each containing the same control word encrypted by an encryption algorithm with respective keys, which are different from one another (column 3, l. 13-23). When pirating becomes excessive, the service provider distributes new user card. The decryption circuit for the control words of the new user cards then contains a new encryption key (column 9, l. 18-22).

However there is no teaching or suggestion in Thomson of entitlement management messages (EMMs) providing the service key (P_T) required to decrypt encrypted first keys (C_W) are sent to the secure devices. Further in Thomson, the source signal or the ECMs are not encrypted using a multiple-key or secret-sharing cryptographic algorithm having a plurality of different decrypting keys or shares ($C_i;P_i$) of which any one or any one or more, respectively, is required for decrypting the encrypted source signal or ECMs, respectively.

Thus, neither Birdwell et al. nor Thomson disclose encrypting the source signal or the ECMs using a multiple-key or secret-sharing cryptographic algorithm having a plurality of

different decrypting keys or shares (C_i;P_i) of which any one or any one or more, respectively, is required for decrypting the encrypted source signal or ECMs, respectively, as recited in claim 6.

It is observed that the terms "multiple-key cryptographic algorithm" and "secret-sharing algorithm" have a well-defined meaning in the art. In support of this observation, relevant extracts from the book by Schneier, mentioned on page 8 of the application as filed, are included with this response.

In addition, the subject matter of claim 6 does not follow from a combination of Birdwell et al. and the common general knowledge as represented, for example, by the enclosed extracts (referring to the book by Schneier as mentioned above) because the feature that any one of the different decrypting keys in a multiple-key cryptographic algorithm or any one or more of the different shares in a secret-sharing cryptographic algorithm are required for decrypting the encrypted source signal or ECMs is not suggested by the prior art. This feature allows the provision of different EMMs for tracing illicit key sharers without causing black-outs. The fact that such an effect had not been achieved prior to the effective date of claim 6 lends support to finding the subject matter included in claim 6 to be non-obvious and thus patentable.

Because neither Birdwell et al. nor Thomson, either alone or in combination, teaches or suggests each of the elements included in claim 6, claim 6 is not obvious in view of the proposed combination of Birdwell et al. and Thomson. Therefore, the 35 U.S.C. § 103(a) rejection of claim 6 cannot stand.

Claim 9 depends from claim 6, and so includes all of the elements recited in claim 6. For at least the reasons states above with respect to claim 6, Birdwell et al. fails to teach each of he elements included in claim 9. Therefore, claim 9 is also not obvious in view of the proposed combination of Birdwell et al. and Thomson. Thus, the 35 U.S.C. § 103(a) rejection of claim 9 cannot stand.

The Final Office Action fails to show how the prior art teaches or suggests making the proposed combination of Birdwell et al. and Thomson with a reasonable expectation of success.

The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, not in applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991); MPEP § 2143. The Examiner must avoid hindsight. *In re Bond*, 910 F.2d 831, 834, 15 USPQ2d 1566, 1568 (Fed. Cir. 1990).

There is no suggestion, motivation or incentive for the skilled person to combine the teachings of Birdwell et al. and Thomson, since they disclose different ways of distributing the authorization keys and encryption keys, respectively. Birdwell et al. uses messages, whereas in Thomson a user card is replaced. The Final Office Action fails to explain how "combining the tracking system of Birdwell et al. into that of Thomson Multimedia for the advantages of improved system security"[1] would be achieved with a reasonable expectation of success, given the above mentioned differences between Birdwell et al. and Thomson.

By failing to meet the above stated requirements, the Final Office Action fails to meet the requirements for forming a rejection of claims 6 and 9 based on the proposed combination of Birdwell et al. and Thomson.

For at least the reasons stated above, Applicants respectfully request withdrawal of the rejection, and reconsideration and allowance of claims 6 and 9.

Allowable Subject Matter

Claims 1-3, 5, and 7 are allowed. Applicants acknowledge the allowance of claims 1-3, 5, and 7.

Reservation of Rights

Applicants do not admit that references cited under 35 U.S.C. §§ 102(a), 102(e), 103/102(a), or 103/102(e) are prior art, and reserve the right to swear behind them at a later date. Arguments presented to distinguish such references should not be construed as admissions that the references are prior art.

[1] *See* the Final Office Action on page 4 at item number 15.

CONCLUSION

Applicants respectfully submit that the claims are in condition for allowance and notification to that effect is earnestly requested. The Examiner is invited to telephone Applicant's attorney 408-278-4042 to facilitate prosecution of this application.

If necessary, please charge any additional fees or credit overpayment to Deposit Account No. 19-0743.

Respectfully submitted,

ANDREW AUGUSTINE WAJS ET AL.

By their Representatives,

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. Box 2938
Minneapolis, MN 55402
408-278-4042

Date 07/05/06

By _____
Andre L. Marais
Reg. No. 48,095

CERTIFICATE UNDER 37 CFR 1.8: The undersigned hereby certifies that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail, in an envelope addressed to: Mail Stop AF, Commissioner of Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on this 5 day of April, 2006.

Dawn R. Shaw — Dawn R Shaw

Name — Signature

APPLIED CRYPTOGRAPHY, SECOND EDITION

PROTOCOLS, ALGORITHMS, AND SOURCE CODE IN C

John Wiley & Sons, Inc.
New York • Chichester • Brisbane • Toronto • Singapore

Publisher: Katherine Schowalter
Editor: Phil Sutherland
Assistant Editor: Allison Roarty
Managing Editor: Robert Aronds
Text Design & Composition: North Market Street Graphics

This text is printed on acid-free paper.

Library of Congress Cataloging-in-Publication Data:

Schneier, Bruce
Applied Cryptography Second Edition : protocols, algorithms, and source code in C / Bruce Schneier.
p. cm.
Includes bibliographical references (p. 675).
ISBN 0-471-12845-7 (cloth : acid-free paper). — ISBN 0-471-11709-9 (paper : acid-free paper)
1. Computer security. 2. Telecommunication—Security measures. 3. Cryptography. I. Title.
QA76.9.A25S35 1996
005.8'2—dc20 95-12398
CIP

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

From these actions, requirements can be specified. For example:

— If Bob accepted message M from Alice at some point in the past, then Eve did not learn M at some point in the past.

— If Bob accepted message M from Alice in Bob's local round N, then Alice sent M to Bob as a response to a query in Bob's local round N.

To use the NRL Protocol Analyzer, a protocol must be specified using the previous constructs. Then, there are four phases of analysis: defining transition rules for honest participants, describing operations available to all—honest and dishonest—participants, describing the basic building blocks of the protocol, and describing the reduction rules. The point of all this is to show that a given protocol meets its requirements. Tools like the NRL Protocol Analyzer could eventually lead to a protocol that can be proven secure.

While much of the work in formal methods involves applying the methods to existing protocols, there is some push towards using formal methods to design the protocols in the first place. Some preliminary steps in this direction are [711]. The NRL Protocol Analyzer also attempts to do this [1512,222,1513].

The application of formal methods to cryptographic protocols is still a fairly new idea and it's really hard to figure out where it is headed. At this point, the weakest link seems to be the formalization process.

3.5 MULTIPLE-KEY PUBLIC-KEY CRYPTOGRAPHY

Public-key cryptography uses two keys. A message encrypted with one key can be decrypted with the other. Usually one key is private and the other is public. However, let's assume that Alice has one key and Bob has the other. Now Alice can encrypt a message so that only Bob can decrypt it, and Bob can encrypt a message so that only Alice can read it.

This concept was generalized by Colin Boyd [217]. Imagine a variant of public-key cryptography with three keys: K_A, K_B, and K_C, distributed as shown in Table 3.2.

Alice can encrypt a message with K_A so that Ellen, with K_B and K_C, can decrypt it. So can Bob and Carol in collusion. Bob can encrypt a message so that Frank can read

TABLE 3.2
Three-Key Key Distribution

Alice	K_A
Bob	K_B
Carol	K_C
Dave	K_A and K_B
Ellen	K_B and K_C
Frank	K_C and K_A

it, and Carol can encrypt a message so that Dave can read it. Dave can encrypt a message with K_A so that Ellen can read it, with K_B so that Frank can read it, or with both K_A and K_B so that Carol can read it. Similarly, Ellen can encrypt a message so that either Alice, Dave, or Frank can read it. All the possible combinations are summarized in Table 3.3; there are no other ones.

This can be extended to n keys. If a given subset of the keys is used to encrypt the message, then the other keys are required to decrypt the message.

Broadcasting a Message

Imagine that you have 100 operatives out in the field. You want to be able to send messages to subsets of them, but don't know which subsets in advance. You can either encrypt the message separately for each person or give out keys for every possible combination of people. The first option requires a lot of messages; the second requires a lot of keys.

Multiple-key cryptography is much easier. We'll use three operatives: Alice, Bob, and Carol. You give Alice K_A and K_B, Bob K_B and K_C, and Carol K_C and K_A. Now you can talk to any subset you want. If you want to send a message so that only Alice can read it, encrypt it with K_C. When Alice receives the message, she decrypts it with K_A and then K_B. If you want to send a message so that only Bob can read it, encrypt it with K_A; so that only Carol can read it, with K_B. If you want to send a message so that both Alice and Bob can read it, encrypt it with K_A and K_C, and so on.

This might not seem exciting, but with 100 operatives it is quite efficient. Individual messages mean a shared key with each operative (100 keys total) and each message. Keys for every possible subset means $2^{100} - 2$ different keys (messages to all operatives and messages to no operatives are excluded). This scheme needs only one encrypted message and 100 different keys. The drawback of this scheme is that you also have to broadcast which subset of operatives can read the message, otherwise each operative would have to try every combination of possible keys looking for the correct one. Even just the names of the intended recipients may be significant. At least for the straightforward implementation of this, everyone gets a really large amount of key data.

There are other techniques for message broadcasting, some of which avoid the previous problem. These are discussed in Section 22.7.

TABLE 3.3
Three-Key Message Encryption

Encrypted with Keys:	Must be Decrypted with Keys:
K_A	K_B and K_C
K_B	K_A and K_C
K_C	K_A and K_B
K_A and K_B	K_C
K_A and K_C	K_B
K_B and K_C	K_A

This is an adjudicated protocol. Trent has absolute power and can do whatever he wants. He can hand out gibberish and claim that it is a valid piece of the secret; no one will know it until they try to reconstruct the secret. He can hand out a piece to Alice, Bob, Carol, and Dave, and later tell everyone that only Alice, Carol, and Dave are needed to reconstruct the secret, and then fire Bob. But since this is Trent's secret to divide up, this isn't a problem.

However, this protocol has a problem: If any of the pieces gets lost and Trent isn't around, so does the message. If Carol, who has a piece of the sauce recipe, goes to work for the competition and takes her piece with her, the rest of them are out of luck. She can't reproduce the recipe, but neither can Alice, Bob, and Dave working together. Her piece is as critical to the message as every other piece combined. All Alice, Bob, or Dave know is the length of the message—nothing more. This is true because R, S, T, U, and M all have the same length; seeing anyone of them gives the length of M. Remember, M isn't being split in the normal sense of the word; it is being XORed with random values.

3.7 Secret Sharing

You're setting up a launch program for a nuclear missile. You want to make sure that no single raving lunatic can initiate a launch. You want to make sure that no two raving lunatics can initiate a launch. You want at least three out of five officers to be raving lunatics before you allow a launch.

This is easy to solve. Make a mechanical launch controller. Give each of the five officers a key and require that at least three officers stick their keys in the proper slots before you'll allow them to blow up whomever we're blowing up this week. (If you're really worried, make the slots far apart and require the officers to insert the keys simultaneously—you wouldn't want an officer who steals two keys to be able to vaporize Toledo.)

We can get even more complicated. Maybe the general and two colonels are authorized to launch the missile, but if the general is busy playing golf then five colonels are required to initiate a launch. Make the launch controller so that it requires five keys. Give the general three keys and the colonels one each. The general together with any two colonels can launch the missile; so can the five colonels. However, a general and one colonel cannot; neither can four colonels.

A more complicated sharing scheme, called a **threshold scheme**, can do all of this and more—mathematically. At its simplest level, you can take any message (a secret recipe, launch codes, your laundry list, etc.) and divide it into n pieces, called **shadows** or **shares**, such that any m of them can be used to reconstruct the message. More precisely, this is called an **(m,n)-threshold scheme**.

With a (3,4)-threshold scheme, Trent can divide his secret sauce recipe among Alice, Bob, Carol, and Dave, such that any three of them can put their shadows together and reconstruct the message. If Carol is on vacation, Alice, Bob, and Dave can do it. If Bob gets run over by a bus, Alice, Carol, and Dave can do it. However, if Bob gets run over by a bus while Carol is on vacation, Alice and Dave can't reconstruct the message by themselves.

General threshold schemes are even more versatile. Any sharing scenario you can imagine can be modeled. You can divide a message among the people in your building so that to reconstruct it, you need seven people from the first floor and five people from the second floor, unless there is someone from the third floor involved, in which case you only need that person and three people from the first floor and two people from the second floor, unless there is someone from the fourth floor involved, in which case you need that person and one person from the third floor, or that person and two people from the first floor and one person from the second floor, unless there is . . . well, you get the idea.

This idea was invented independently by Adi Shamir [1414] and George Blakley [182] and studied extensively by Gus Simmons [1466]. Several different algorithms are discussed in Section 23.2.

Secret Sharing with Cheaters

There are many ways to cheat with a threshold scheme. Here are just a few of them.

Scenario 1: Colonels Alice, Bob, and Carol are in a bunker deep below some isolated field. One day, they get a coded message from the president: "Launch the missiles. We're going to eradicate the last vestiges of neural network research in the country." Alice, Bob, and Carol reveal their shadows, but Carol enters a random number. She's actually a pacifist and doesn't want the missiles launched. Since Carol doesn't enter the correct shadow, the secret they recover is the wrong secret. The missiles stay in their silos. Even worse, no one knows why. Alice and Bob, even if they work together, cannot prove that Carol's shadow is invalid.

Scenario 2: Colonels Alice and Bob are sitting in the bunker with Mallory. Mallory has disguised himself as a colonel and none of the others is the wiser. The same message comes in from the president, and everyone reveals their shadows. "Bwa-ha-ha!" shouts Mallory. "I faked that message from the president. Now I know both of your shadows." He races up the staircase and escapes before anyone can catch him.

Scenario 3: Colonels Alice, Bob, and Carol are sitting in the bunker with Mallory, who is again disguised. (Remember, Mallory doesn't have a valid shadow.) The same message comes in from the president and everyone reveals their shadows. Mallory reveals his shadow only after he has heard the other three. Since only three shadows are needed to reconstruct the secret, he can quickly create a valid shadow and reveals that. Now, not only does he know the secret, but no one realizes that he isn't part of the scheme.

Some protocols that handle these sorts of cheaters are discussed in Section 23.2.

Secret Sharing without Trent

A bank wants its vault to open only if three out of five officers enter their keys. This sounds like a basic (3,5)-threshold scheme, but there's a catch. No one is to know the entire secret. There is no Trent to divide the secret up into five pieces. There are protocols by which the five officers can create a secret and each get a piece, such that none of the officers knows the secret until they all reconstruct it. I'm not going to discuss these protocols in this book; see [756] for details.

Sharing a Secret without Revealing the Shares

These schemes have a problem. When everyone gets together to reconstruct their secret, they reveal their shares. This need not be the case. If the shared secret is a private key (to a digital signature, for example), then n shareholders can each complete a partial signature of the document. After the nth partial signature, the document has been signed with the shared private key and none of the shareholders learns any other shares. The point is that the secret can be reused, and you don't need a trusted processor to handle it. This concept is explored further by Yvo Desmedt and Yair Frankel [483,484].

Verifiable Secret Sharing

Trent gives Alice, Bob, Carol, and Dave each a share or at least he says he does. The only way any of them know if they have a valid share is to try to reconstruct the secret. Maybe Trent sent Bob a bogus share or Bob accidentally received a bad share through communications error. Verifiable secret sharing allows each of them to individually verify that they have a valid share, without having to reconstruct the secret [558,1235].

Secret-Sharing Schemes with Prevention

A secret is divided up among 50 people so that any 10 can get together and reconstruct the secret. That's easy. But, can we implement the same secret-sharing scheme with the added constraint that 20 people can get together and *prevent* the others from reconstructing the secret, no matter how many of them there are? As it turns out, we can [153].

The math is complicated, but the basic idea is that everyone gets two shares: a "yes" share and a "no" share. When it comes time to reconstruct the secret, people submit one of their shares. The actual share they submit depends on whether they wish the secret reconstructed. If there are m or more "yes" shares and fewer than n "no" shares, the secret can be reconstructed. Otherwise, it cannot.

Of course, nothing prevents a sufficient number of "yes" people from going off in a corner without the "no" people (assuming they know who they are) and reconstructing the secret. But in a situation where everyone submits their shares into a central computer, this scheme will work.

Secret Sharing with Disenrollment

You've set up your secret-sharing system and now you want to fire one of your shareholders. You could set up a new scheme without that person, but that's time-consuming. There are methods for coping with this system. They allow a new sharing scheme to be activated instantly once one of the participants becomes untrustworthy [1004].

3.8 Cryptographic Protection of Databases

The membership database of an organization is a valuable commodity. On the one hand, you want to distribute the database to all members. You want them to com-

CHAPTER 23

Special Algorithms for Protocols

23.1 MULTIPLE-KEY PUBLIC-KEY CRYPTOGRAPHY

This is a generalization of RSA (see Section 19.3) [217,212]. The modulus, n, is the product of two primes, p and q. However, instead of choosing e and d such that $ed \equiv 1 \bmod (p-1)(q-1)$, choose t keys, K_i, such that

$$K_1 * K_2 * \ldots * K_t \equiv 1 \bmod (p-1)(q-1)$$

Since

$$M^{K_1 \cdot K_2 \cdot \ldots \cdot K_t} = M$$

this is a multiple-key scheme as described in Section 3.5.

If, for example, there are five keys, a message encrypted with K_3 and K_5 can be decrypted with K_1, K_2, and K_4:

$$C = M^{K_3 \cdot K_5} \bmod n$$
$$M = C^{K_1 \cdot K_2 \cdot K_4} \bmod n$$

One use for this is multisignatures. Imagine a situation where both Alice and Bob have to sign a document for it to be valid. Use three keys: K_1, K_2, and K_3. The first two are issued one each to Alice and Bob, and the third is made public.

(1) First Alice signs M and sends it to Bob.

$$M' = M^{K_1} \bmod n$$

(2) Bob can recover M from M'.

$$M = M'^{K_2 \cdot K_3} \bmod n$$

(3) He can also add his signature.

$$M'' = M'^{K_2} \bmod n$$

(4) Anyone can verify the signature with K_3, the public key.

$$M = M''^{K_3} \bmod n$$

Note that a trusted party is needed to set this system up and distribute the keys to Alice and Bob. Another scheme with the same problem is [484]. Yet a third scheme is [695,830,700], but the effort in verification is proportional to the number of signers. Newer schemes [220,1200] based on zero-knowledge identification schemes solve both shortcomings of the previous systems.

23.2 SECRET-SHARING ALGORITHMS

Back in Section 3.7 I discussed the idea behind secret-sharing schemes. The four different algorithms that follow are all particular cases of a general theoretical framework [883].

LaGrange Interpolating Polynomial Scheme

Adi Shamir uses polynomial equations in a finite field to construct a threshold scheme [1414]. Choose a prime, p, which is both larger than the number of possible shadows and larger than the largest possible secret. To share a secret, generate an arbitrary polynomial of degree $m - 1$. For example, if you want to create a $(3,n)$-threshold scheme (three shadows are necessary to reconstruct M), generate a quadratic polynomial

$$(ax^2 + bx + M) \bmod p$$

where p is a random prime larger than any of the coefficients. The coefficients a and b are chosen randomly; they are kept secret and discarded after the shadows are handed out. M is the message. The prime must be made public.

The shadows are obtained by evaluating the polynomial at n different points:

$$k_i = F(x_i)$$

In other words, the first shadow could be the polynomial evaluated at $x = 1$, the second shadow could be the polynomial evaluated at $x = 2$, and so forth.

Since the quadratic polynomial has three unknown coefficients, a, b, and M, any three shadows can be used to create three equations. Two shadows cannot. One shadow cannot. Four or five shadows are redundant.

For example, let M be 11. To construct a (3, 5)-threshold scheme, where any three of five people can reconstruct M, first generate a quadratic equation (7 and 8 were chosen randomly):

$$F(x) = (7x^2 + 8x + 11) \bmod 13$$

The five shadows are:

$$k_1 = F(1) = 7 + 8 + 11 \equiv 0 \pmod{13}$$
$$k_2 = F(2) = 28 + 16 + 11 \equiv 3 \pmod{13}$$
$$k_3 = F(3) = 63 + 24 + 11 \equiv 7 \pmod{13}$$

$$k_4 = F(4) = 112 + 32 + 11 \equiv 12 \pmod{13}$$
$$k_5 = F(5) = 175 + 40 + 11 \equiv 5 \pmod{13}$$

To reconstruct M from three of the shadows, for example k_2, k_3, and k_5, solve the set of linear equations:

$$a * 2^2 + b * 2 + M \equiv 3 \pmod{13}$$
$$a * 3^2 + b * 3 + M \equiv 7 \pmod{13}$$
$$a * 5^2 + b * 5 + M \equiv 5 \pmod{13}$$

The solution will be $a = 7$, $b = 8$, and $M = 11$. So M is recovered.

This sharing scheme can be easily implemented for larger numbers. If you want to divide the message into 30 equal parts such that any six can get together and reproduce the message, give each of the 30 people the evaluation of a polynomial of degree 5.

$$F(x) = ax^6 + bx^5 + cx^4 + dx^3 + ex^2 + fx + M \pmod{p}$$

Six people can solve for the six unknowns (including M); five people cannot learn anything about M.

The most mind-boggling aspect of secret sharing is that if the coefficients are picked randomly, five people with infinite computing power can't learn anything more than the length of the message (which each of them knows anyway). This is as secure as a one-time pad; an attempt at exhaustive search (that is, trying all possible sixth shadows) will reveal that any conceivable message could be the secret. This is true for all the secret-sharing schemes presented here.

Vector Scheme

George Blakley invented a scheme using points in space [182]. The message is defined as a point in m-dimensional space. Each shadow is the equation of an $(m - 1)$-dimensional hyperplane that includes the point. The intersection of any m of the hyperplanes exactly determines the point.

For example, if three shadows are required to reconstruct the message, then it is a point in three-dimensional space. Each shadow is a different plane. With one shadow, you know the point is somewhere on the plane. With two shadows, you know the point is somewhere on the line formed where the two planes intersect. With three shadows, you can determine the point exactly: the intersection of the three planes.

Asmuth-Bloom

This scheme uses prime numbers [65]. For an (m, n)-threshold scheme, choose a large prime, p, greater than M. Then choose n numbers less than p, $d_1, d_2, \ldots, d_n$, such that:

1. The d values are in increasing order; $d_i < d_{i+1}$
2. Each d_i is relatively prime to every other d_i
3. $d_1 * d_2 * \ldots * d_m > p * d_{n-m+2} * d_{n-m+3} * \ldots * d_n$

To distribute the shadows, first choose a random value r and compute

$$M' = M + rp$$

The shadows, k_i, are

$$k_i = M' \bmod d_i$$

Any m shadows can get together and reconstruct M using the Chinese remainder theorem, but any $m - 1$ cannot. See [65] for details.

Karnin-Greene-Hellman

This scheme uses matrix multiplication [818]. Choose $n + 1$ m-dimensional vectors, $V_0, V_1, \ldots, V_n$, such that any possible $m * m$ matrix formed out of those vectors has rank m. The vector U is a row vector of dimension $m + 1$.

M is the matrix product $U \cdot V_0$. The shadows are the products $U \cdot V_i$, where i is a number from 1 to n.

Any m shadows can be used to solve the $m * m$ system of linear equations, where the unknowns are the coefficients of U. UV_0 can be computed from U. Any $m - 1$ shadows cannot solve the system of linear equations and therefore cannot recover the secret.

Advanced Threshold Schemes

The previous examples illustrate only the simplest threshold schemes: Divide a secret into n shadows such that any m can be used to recover the secret. These algorithms can be used to create far more complicated schemes. The following examples will use Shamir's algorithm, although any of the others will work.

To create a scheme in which one person is more important than another, give that person more shadows. If it takes five shadows to recreate a secret and one person has three shadows while everyone else has only one, then that person and two other people can recreate the secret. Without that person, it takes five to recreate the secret.

Two or more people could get multiple shadows. Each person could have a different number of shadows. No matter how the shadows are distributed, any m of them can be used to reconstruct the secret. Someone with $m - 1$ shadows, be it one person or a roomful of people, cannot do it.

In other types of schemes, imagine a scenario with two hostile delegations. You can share the secret so that two people from the 7 in Delegation A and 3 people from the 12 in Delegation B are required to reconstruct the secret. Make a polynomial of degree 3 that is the product of a linear expression and a quadratic expression. Give everyone from Delegation A a shadow that is the result of an evaluation of the linear equation; give everyone from Delegation B a shadow that is the evaluation of the quadratic equation.

Any two shadows from Delegation A can be used to reconstruct the linear equation, but no matter how many other shadows the group has, they cannot get any information about the secret. The same is true for Delegation B: They can get three shadows together to reconstruct the quadratic equation, but they cannot get any

more information necessary to reconstruct the secret. Only when the two delegations share their equations can they be multiplied to reconstruct the secret.

In general, any type of sharing scheme that can be imagined can be implemented. All you have to do is to envision a system of equations that corresponds to the particular scheme. Some excellent papers on generalized secret-sharing schemes are [1462,1463,1464].

Sharing a Secret with Cheaters

This algorithm modifies the standard (m, n)-threshold scheme to detect cheaters [1529]. I demonstrate this using the LaGrange scheme, although it works with the others as well.

Choose a prime, p, that is both larger than n and larger than

$$(s-1)(m-1)/e+m$$

where s is the largest possible secret and e is the probability of successful cheating. You can make e as small as you want; it just makes the computation more complex. Construct your shadows as before, except instead of using 1, 2, 3, . . . , n for x_i, choose random numbers between 1 and $p-1$ for x_i.

Now, when Mallory sneaks into the secret reconstruction meeting with his false share, his share has a high probability of not being possible. An impossible secret is, of course, a fake secret. See [1529] for the math.

Unfortunately, while Mallory is exposed as a cheater, he still learns the secret (assuming that there are m other valid shares). Another protocol, from [1529,975], prevents that. The basic idea is to have a series of k secrets, such that none of the participants knows beforehand which is correct. Each secret is larger than the one before, except for the real secret. The participants combine their shadows to generate one secret after the other, until they create a secret that is less than the previous secret. That's the correct one.

This scheme will expose cheaters early, before the secret is generated. There are complications when the participants deliver their shadows one at a time; refer to the papers for details. Other papers on the detection and prevention of cheaters in threshold schemes are [355,114,270].

23.3 Subliminal Channel

Ong-Schnorr-Shamir

This subliminal channel (see Section 4.2), designed by Gustavus Simmons [1458,1459,1460], uses the Ong-Schnorr-Shamir identification scheme (see Section 20.5). As in the original scheme, the sender (Alice) chooses a public modulus, n, and a private key, k, such that n and k are relatively prime. Unlike the original scheme, k is shared between Alice and Bob, the recipient of the subliminal message.

The public key is calculated:

$$h = -k^2 \bmod n$$